

List<T>

List<T>

- **List<T>** yapısı sıralama, arama ve liste işlemlerinin gerçekleştirilebildiği koleksiyonlardır.
- **List<T>** **ICollection<T>** uygulayan **ArrayList** yapısının eşdeğeridir.
- **System.Collections.Generic** alan adının altından gelir.

List<T>

- **List<T>** belirtilen türdeki elemanları içerebilir.
- Derleme zamanında tür denetimi sağlanır ve **generic** türde olduğu için kutulama (boxing) - kutunda çıkarma (unboxing) yapılmaz.
- Liste elemanları **Add()** veya **AddRange()** metotları ile ya da tanımlama esnasında oluşturulabilir.

List<T>

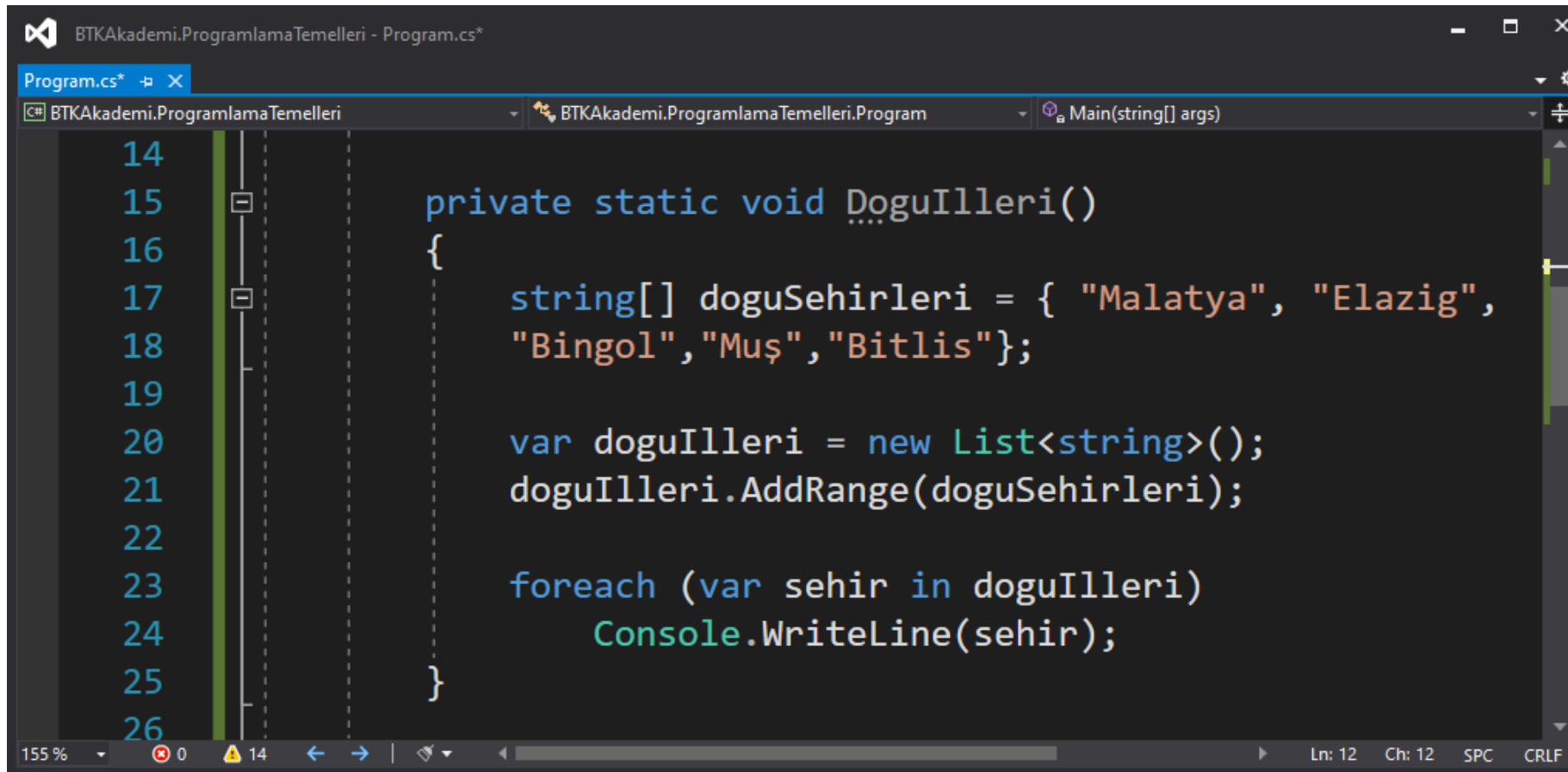
- Indeks değerleri kullanılarak (`myList[0]` gibi) liste üzerindeki elemanlara erişim sağlanabilir.
- `List<T> ArrayList`'e göre daha hızlı ve daha az hata eğilimi gösterir.
- Indeks değerleri kullanılarak (`myList[0]` gibi) liste üzerindeki elemanlara erişim sağlanabilir.
- `List<T> ArrayList`'e göre daha hızlı ve daha az hata eğilimi gösterir.

```
BTAKademi.ProgramlamaTemelleri
Program.cs
BTAKademi.ProgramlamaTemelleri.Program
Main(string[] args)

8  class Program
9  {
10     static void Main(string[] args)
11     {
12         List<int> sayilar = new List<int>();
13         for (int i = 0; i < 10; i++)
14         {
15             sayilar.Add(new Random().Next(1, 9));
16             Console.WriteLine(sayilar[i]);
17         }
18
19         var sehirler = new List<string>();
20         sehirler.Add("Sinop");
21         sehirler.Add("Zonguldak");
22         sehirler.Add("Corum");
23         sehirler.Add("Elazig");
24         sehirler.Add(null); // null degerlerde kabul edilir.
25
26         var buyukSehirler = new List<string>()
27         {
28             "Ankara",
29             "İstanbul",
30             "Samsun"
31         };
32     }
}
```

Liste Oluşturma

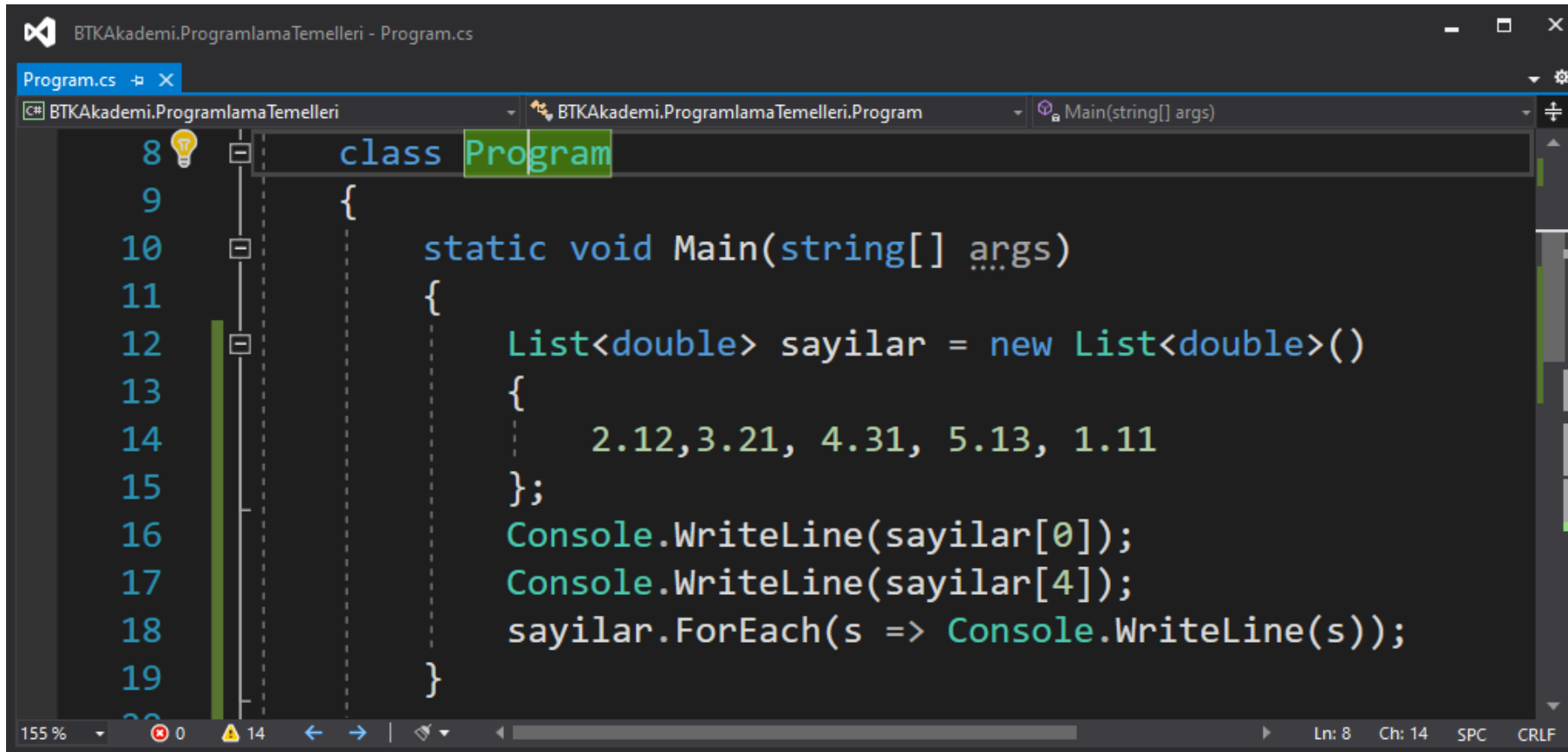
Liste Oluşturma



```
14  
15 private static void DoguIlleri()  
16 {  
17     string[] doguSehirleri = { "Malatya", "Elazig",  
18     "Bingol", "Muş", "Bitlis"};  
19  
20     var doguIlleri = new List<string>();  
21     doguIlleri.AddRange(doguSehirleri);  
22  
23     foreach (var sehir in doguIlleri)  
24         Console.WriteLine(sehir);  
25 }  
26
```

155 % 0 14 Ln: 12 Ch: 12 SPC CRLF

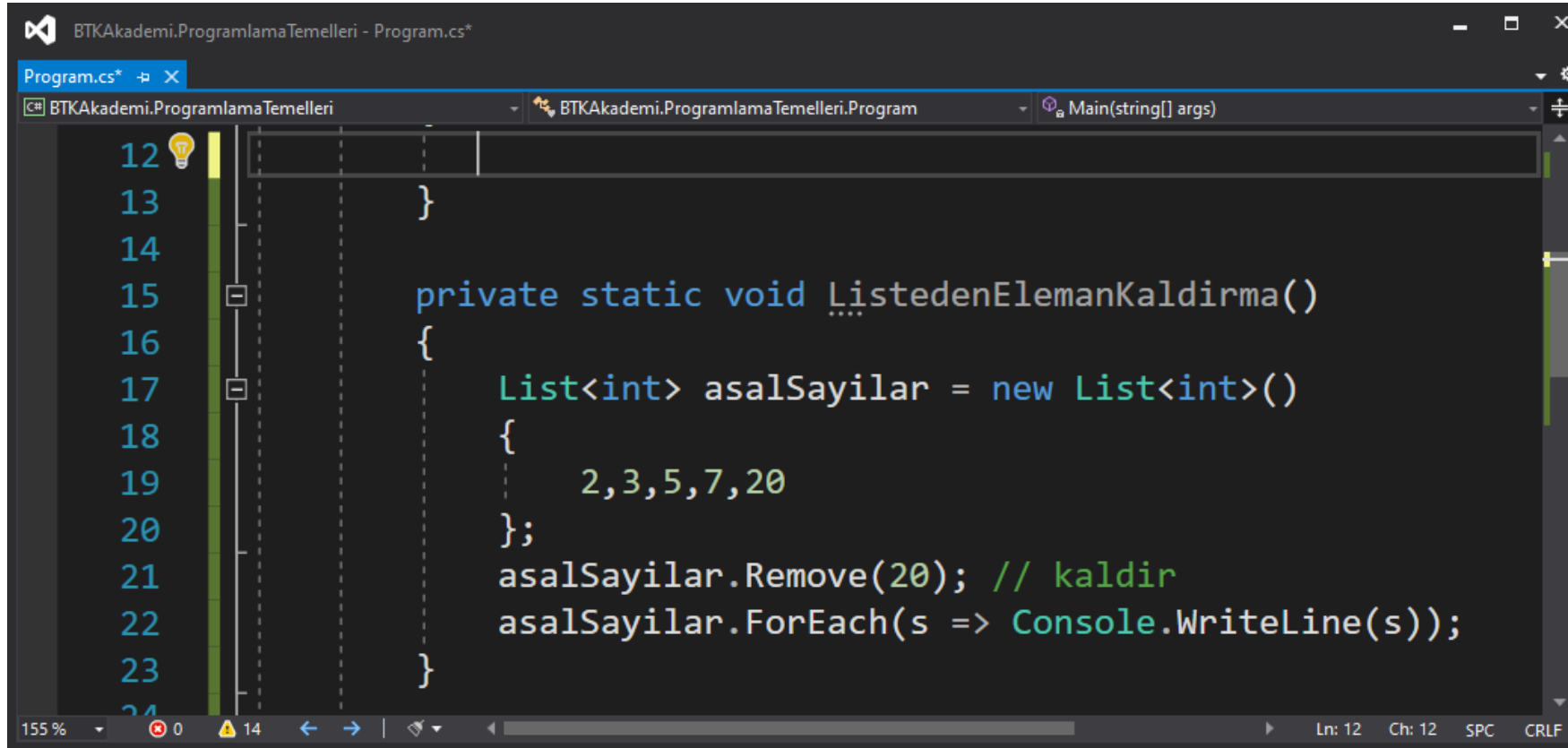
Liste Oluşturma ve Liste Elemanlarına Erişme



```
8 class Program
9 {
10     static void Main(string[] args)
11     {
12         List<double> sayilar = new List<double>()
13         {
14             2.12, 3.21, 4.31, 5.13, 1.11
15         };
16         Console.WriteLine(sayilar[0]);
17         Console.WriteLine(sayilar[4]);
18         sayilar.ForEach(s => Console.WriteLine(s));
19     }
20 }
```

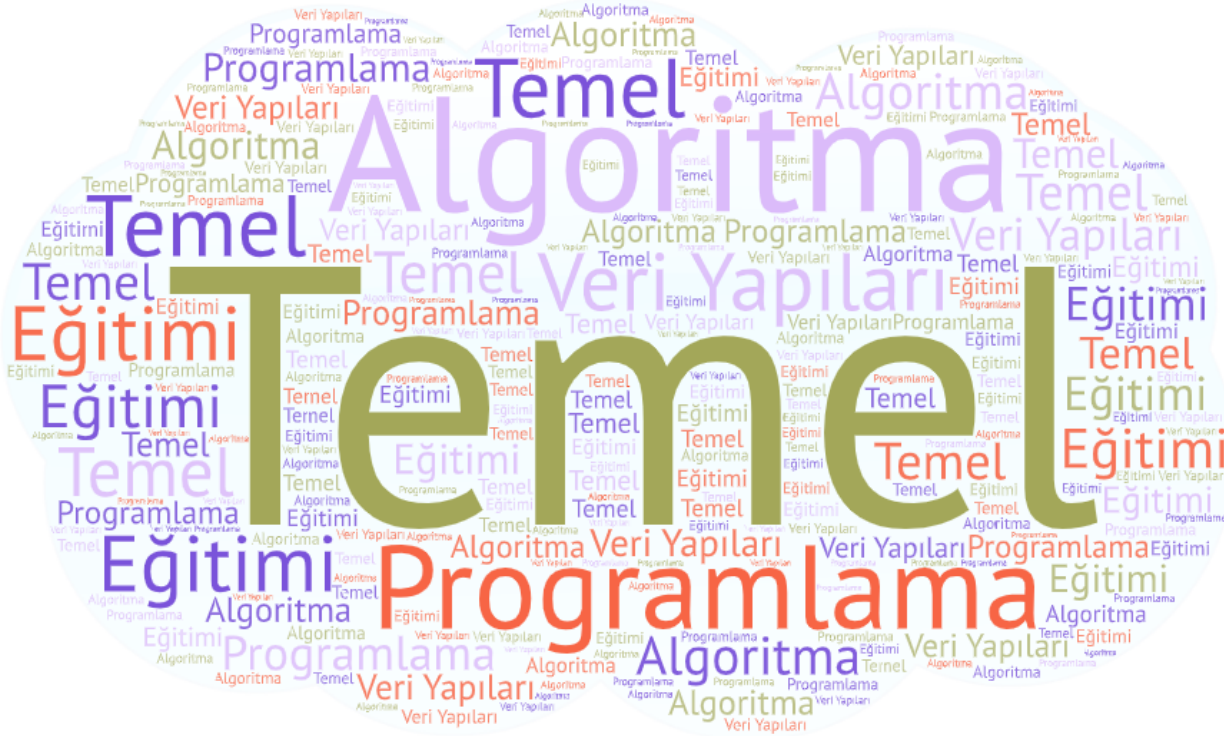
The screenshot shows a Visual Studio Code editor window titled "BTKAkademi.ProgramlamaTemelleri - Program.cs". The code defines a class `Program` with a static `Main` method. Inside `Main`, a `List<double>` named `sayilar` is created and populated with five values: 2.12, 3.21, 4.31, 5.13, and 1.11. The code then prints the first element (`sayilar[0]`), the fifth element (`sayilar[4]`), and iterates over all elements using `ForEach` to print each one. The status bar at the bottom indicates the cursor is at line 8, column 14.

Listeden eleman kaldırma



```
12 }
13
14
15 private static void ListedenElemanKaldirma()
16 {
17     List<int> asalSayilar = new List<int>()
18     {
19         2,3,5,7,20
20     };
21     asalSayilar.Remove(20); // kaldır
22     asalSayilar.ForEach(s => Console.WriteLine(s));
23 }
```

The screenshot shows a Visual Studio Code editor window titled "BTKAkademi.ProgramlamaTemelleri - Program.cs*". The editor displays a C# program with a method named `ListedenElemanKaldirma()`. Inside this method, a `List<int>` named `asalSayilar` is created and initialized with the values 2, 3, 5, 7, and 20. The `Remove(20)` method is called to remove the element 20 from the list. Finally, the `ForEach` method is used to iterate over the list and print each element to the console using `Console.WriteLine(s)`. The status bar at the bottom indicates the cursor is at line 12, column 12, with a 155% zoom level.



List<T>